Administracion de Base de Datos

Ejemplo de Base de Datos de una Universidad

Tablas

Students (Estudiantes): Almacena información sobre los estudiantes.

Courses (Cursos): Almacena información sobre los cursos ofrecidos.

Professors (Profesores): Almacena información sobre los profesores.

Enrollments (Matrículas): Almacena información sobre qué estudiantes están matriculados en qué cursos.

PostgresSQL

```
CREATE TABLE Students (
    student_id SERIAL PRIMARY KEY,
    name VARCHAR(100),
    email VARCHAR(100)
);

CREATE TABLE Courses (
    course_id SERIAL PRIMARY KEY,
    title VARCHAR(100),
    department VARCHAR(100),
    professor_id INT,
    FOREIGN KEY (professor_id) REFERENCES Professors (professor_id)
);

CREATE TABLE Professors (
    professor_id SERIAL PRIMARY KEY,
    name VARCHAR(100),
    email VARCHAR(100)
);

CREATE TABLE Enrollments (
    enrollment_id SERIAL PRIMARY KEY,
    student_id INT,
    course_id INT,
    FOREIGN KEY (student_id) REFERENCES Students (student_id),
    FOREIGN KEY (course_id) REFERENCES Courses (course_id)
);

CREATE VIEW EnrollmentDetails AS
SELECT e.enrollment_id, s.name AS student_name, c.title AS course_title
FROM Enrollments e
JOIN Students s ON e.student_id = s.student_id
JOIN Courses c ON e.course_id = c.course_id;
```

Mysql

```
CREATE TABLE Students (
    student_id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(100),
    email VARCHAR(100)
);

CREATE TABLE Courses (
    course_id INT AUTO_INCREMENT PRIMARY KEY,
    title VARCHAR(100),
    department VARCHAR(100),
    professor_id INT,
    FOREIGN KEY (professor_id) REFERENCES Professors (professor_id)
);

CREATE TABLE Professors (
    professor_id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(100),
    email VARCHAR(100)
);

CREATE TABLE Enrollments (
    enrollment_id INT AUTO_INCREMENT PRIMARY KEY,
    student_id INT,
    course_id INT,
    FOREIGN KEY (student_id) REFERENCES Students (student_id),
    FOREIGN KEY (course_id) REFERENCES Courses (course_id)
);

CREATE VIEW EnrollmentDetails AS
SELECT e.enrollment_id, s.name AS student_name, c.title AS course_title
FROM Enrollments e
JOIN Students s ON e.student_id = s.student_id
JOIN Courses c ON e.course_id = c.course_id;
```

Procedimientos Almacenados

PstgresSQL

Supongamos que queremos crear un procedimiento almacenado para inscribir a un estudiante en un curso específico.

```
CREATE OR REPLACE FUNCTION EnrollStudent(student_name VARCHAR, course_title VARCHAR)
RETURNS VOID AS $$
DECLARE
    student_id INT;
    course_id INT;
BEGIN
    SELECT student_id INTO student_id FROM Students WHERE name = student_name;
    SELECT course_id INTO course_id FROM Courses WHERE title = course_title;
    INSERT INTO Enrollments (student_id, course_id) VALUES (student_id, course_id);
END;
$$ LANGUAGE plpgsql;

```

Mysql

```
DELIMITER //

CREATE PROCEDURE EnrollStudent(IN student_name VARCHAR(100), IN course_title
VARCHAR(100))
BEGIN
    DECLARE student_id INT;
    DECLARE course_id INT;

    SELECT student_id INTO student_id FROM Students WHERE name = student_name;
    SELECT course_id INTO course_id FROM Courses WHERE title = course_title;
    INSERT INTO Enrollments (student_id, course_id) VALUES (student_id, course_id);
END //

DELIMITER;
```